Computational Optimization and Applications (2021) 79:339–368
https://doi.org/10.1007/s10589-021-00271-w

A stochastic subspace approach to gradient-free optimization in high dimensions

David Kozak[1,2] · Stephen Becker[3] · Alireza Doostan[4] · Luis Tenorio[1]

Received: 19 February 2020 / Accepted: 1 March 2021 / Published online: 13 April 2021

Abstract
We present a stochastic descent algorithm for unconstrained optimization that is particularly efficient when the objective function is slow to evaluate and gradients are not easily obtained, as in some PDE-constrained optimization and machine learning problems. The algorithm maps the gradient onto a low-dimensional random subspace of dimension ℓ at each iteration, similar to coordinate descent but without restricting directional derivatives to be along the axes. Without requiring a full gradient, this mapping can be performed by computing ℓ directional derivatives (e.g., via forward-mode automatic differentiation). We give proofs for convergence in expectation under various convexity assumptions as well as probabilistic convergence results under strong-convexity. Our method provides a novel extension to the well-known Gaussian smoothing technique to descent in subspaces of dimension greater than one, opening the doors to new analysis of Gaussian smoothing when more than one directional derivative is used at each iteration. We also provide a finite-dimensional variant of a special case of the Johnson–Lindenstrauss lemma. Experimentally, we show that our method compares favorably to coordinate descent, Gaussian smoothing, gradient descent and BFGS (when gradients are calculated via forward-mode automatic differentiation) on problems from the machine learning and shape optimization literature.

Keywords Randomized methods · Gradient-free · Gaussian processes · Stochastic gradients

✉ David Kozak
dkozak@soleaenergy.com

1 Solea Energy, Overland Park, KS, USA

2 Department of Applied Mathematics and Statistics, Colorado School of Mines, Golden, CO, USA

3 Department of Applied Mathematics, University of Colorado, Boulder, CO, USA

4 Aerospace Engineering Sciences Department, University of Colorado, Boulder, CO, USA

Mathematics Subject Classification 90C06 · 93B40 · 65K10

1 Introduction

We consider optimization problems of the form

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}), \tag{1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ has λ-Lipschitz gradient but $\nabla f(\mathbf{x})$ is costly to evaluate. We also consider additional restrictions on f such as convexity or γ-strong convexity, which will be made clear as required. The main idea is straightforward and has a long history: descend along directions in input space rather than along the gradient.

Directional derivatives can be obtained exactly by forward-mode automatic differentiation, as discussed in [53], at a cost of approximately one function evaluation per direction. The gradient can be obtained by performing d such calculations in orthogonal directions. Reverse-mode automatic differentiation would enable calculation of the gradient at a cost of roughly four function evaluations [53] but it has a potential explosion of memory when creating temporary intermediate variables. For example, in unsteady fluid flow, the naive adjoint state method requires storing the entire time-dependent PDE-solution [54]. Hybrid check-pointing schemes [67], designed to reduce memory-overhead, are the subject of active research but the issue has not yet been satisfactorily resolved. We desire methods that can make progress towards the optima after fewer than d function evaluations per iteration, while still providing convergence guarantees similar to those of traditional methods. To this end, we approximate $\nabla f(\mathbf{x}_k)$ with ℓ directional derivatives determined by a random matrix $\mathbf{P}_k \in \mathbb{R}^{d\times\ell}$. Such a choice amounts to descending in an ℓ-dimensional subspace of gradient space and results in the following recursion,

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \mathbf{P}_k \mathbf{P}_k^\top \nabla f(\mathbf{x}_k), \tag{2}$$

where $\alpha > 0$ is fixed, $\mathbf{P}_k \in \mathbb{R}^{d\times\ell}$ is a random matrix with the properties $\mathbb{E}\,\mathbf{P}_k\mathbf{P}_k^\top = \mathbf{I}_d$ and $\mathbf{P}_k^\top\mathbf{P}_k = (d/\ell)\,\mathbf{I}_\ell$. Note that when $\mathbf{P}_k\mathbf{P}_k^\top$ is diagonal (2) reduces to randomized block-coordinate descent. In this document we show that randomized block-coordinate descent is suboptimal for algorithms of the form (2) due to its strong dependence on both the ambient dimension of the problem and the structure of the gradient. Using a variant of the Johnson–Lindenstrauss lemma we provide non-asymptotic, probabilistic convergence results with spherically symmetric random matrices $\mathbf{P}_k$, results that we show do not hold for coordinate descent.

For concreteness consider the matrix $\mathbf{P}$ comprised of columns $\mathbf{P}^1, \dots, \mathbf{P}^\ell$. Then an ℓ-dimensional subspace approximating the gradient can be obtained using finite-differences

$$\nabla f(\mathbf{x}) \approx \mathbf{P}\begin{pmatrix} \frac{f(\mathbf{x}+\mathbf{P}^1 h)-f(\mathbf{x})}{h} \\ \vdots \\ \frac{f(\mathbf{x}+\mathbf{P}^\ell h)-f(\mathbf{x})}{h} \end{pmatrix}. \tag{3}$$

By using exact directional derivatives obtained with forward-mode automatic differentiation, (3) reduces to $\nabla f(\mathbf{x}) \approx \mathbf{P}\mathbf{P}^\top \nabla f(\mathbf{x})$, resulting in the form for (2). In this paper we analyze the effect that the choice of matrices $\mathbf{P}$ can have on the convergence of (2). This is accomplished, in part, by analyzing how well $\mathbf{P}\mathbf{P}^\top \nabla f(\mathbf{x})$ approximates the gradient.

A particular case of (1) is Empirical Risk Minimization (ERM) commonly used in machine learning, where $f(\mathbf{x}) = (1/n)\sum_{i=1}^n f_i(\mathbf{x})$ and n is typically very large. Hence an ERM problem is amenable to iterative stochastic methods that approximate $\nabla f(\mathbf{x})$ using S randomly sampled observations, $(i_s)_{s=1}^S \subset \{1,\dots,n\}$, at each iteration with $f_S(\mathbf{x}) = (1/S)\sum_{s=1}^S f_{i_s}(\mathbf{x})$ where $S \ll n$. While the methods we discuss do not require a finite-sum structure, they can be used for such problems.

There are important classes of functions that do not fit into the ERM framework and therefore do not benefit from stochastic gradient descent which is tailored to ERM. Partial Differential Equation (PDE) constrained optimization is one such example, and except in special circumstances (such as [33]), a stochastic approach leveraging the ERM structure (such as stochastic gradient descent and its variants) does not provide any benefits. This is because in PDE-constrained optimization the cost of evaluating each $\nabla f_i(\mathbf{x})$ is often identical to the cost of evaluating $\nabla f(\mathbf{x})$. Problems outside of the ERM framework are not limited to parameter estimation for PDEs. For example, parameter estimation of Gaussian processes, specifically the sparse Gaussian process framework of [63, 66] does not benefit from an ERM structure but can benefit from our methodology.

PDE-constrained optimization Partial differential equations are frequently used to model physical phenomena. Successful application of PDEs to modeling is contingent upon appropriate discretization and parameter estimation. Parameter estimation in this setting arises in optimal control, or whenever the parameters of the PDE are unknown, as in inverse problems. Algorithmic and hardware advances for PDE-constrained optimization have allowed for previously impossible modeling capabilities. Examples include fluid dynamics models with millions of parameters for tracking atmospheric contaminants [25], modeling the flow of the Antarctic ice sheet [40, 58], parameter estimation in seismic inversion [1, 12], groundwater hydrology [9], experimental design [34, 38], and atmospheric remote sensing [16].

Gaussian processes Gaussian processes are an important class of stochastic processes. In this paper we use them to model an unknown function in the context of regression. The celebrated representer theorem of Kimeldorf and Wahba [42] allows the modeling of functions from an infinite-dimensional reproducing kernel Hilbert space using only machinery from finite-dimensional linear algebra. However, the applications of Gaussian processes are somewhat hamstrung in many modern settings because their time complexity scales as $\mathcal{O}(n^3)$ and their storage as $\mathcal{O}(n^2)$. One recourse is to approximate the Gaussian process, allowing time complexity to be reduced to $\mathcal{O}(nm^2)$ with storage requirements of $\mathcal{O}(nm)$, where $m \ll n$ is the number of points used in lieu of the full data set. Methods have been developed to place these m inducing points, also called landmark points, along the domain at points different from the original inputs [63, 66]; optimal placement of the landmarks is a continuous optimization problem with dimension equal to the number of inducing points to be placed in addition to the number of parameters to be estimated. Such a framework places a great burden on the optimization procedure as improperly placed landmark points may result in poor approximations.

1.1 Related work

Despite being among the easiest to understand and oldest variants of gradient descent, subspace methods (by far the most common of which is coordinate descent) have, until recently, attracted relatively little attention in the optimization literature.

Coordinate descent schemes The simplest variant of subspace descent is a deterministic method that cycles over the coordinates. This method is popular because many problems have structure that makes a coordinate update very cheap. However convergence results for coordinate descent require challenging analysis and the class of functions for which it converges is restricted; indeed, [60, 68] provide simple examples for which the method fails to converge while simpler-to-analyze methods such as gradient descent converge.

Choosing the coordinates randomly can lead to results on par with gradient descent [51, 61]. Much emphasis has been placed recently on accelerating coordinate descent methods [3, 37], but the improvements require knowledge of the Lipschitz constants of the partial derivatives of the functions and/or special structure in the function to make updates inexpensive and to choose a sampling scheme. See [71] for a survey of recent results.

A generalization of coordinate descent for linear systems is provided by [30] wherein the goal is to solve the dual problem. The idea proposed in [30] of descending in a random direction according to some pre-specified distribution that is not uniform makes it more similar to ours than other algorithms that focus on solving the dual problem such as, e.g., [62].

Zeroth-order optimization Our methods use directions $\mathbf{P}^\top \nabla f(\mathbf{x})$, where $\mathbf{P}$ is $d \times \ell$ with $\ell \ll d$, which is equivalent to taking ℓ directional derivatives of f at $\mathbf{x}$. Observe that as our methods do not use gradients they fall into the class of gradient-free optimization, however since we use exact directional derivatives the methods are not derivative-free. To be clear, when $\nabla f(\mathbf{x})$ is readily available, zeroth-order optimization methods are not competitive with first- or second-order methods. For example, if $f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$, with $\mathbf{A} \in \mathbb{R}^{n\times d}$ and $\mathbf{b} \in \mathbb{R}^n$ then evaluating $f(\mathbf{x})$ and evaluating $\nabla f(\mathbf{x}) = 2\mathbf{A}^\top(\mathbf{A}\mathbf{x} - \mathbf{b})$ have nearly the same computational cost, namely $\mathcal{O}(nd)$. In fact, such a statement is true regardless of the structure of f: by using reverse-mode automatic differentiation (AD), one can theoretically evaluate $\nabla f(\mathbf{x})$ in about four-times the cost of evaluating $f(\mathbf{x})$, regardless of the dimension d [31]. In the context of PDE-constrained optimization, the popular adjoint-state method, which is a form of AD applied to either the continuous or discretized PDE, also evaluates $\nabla f(\mathbf{x})$ in time independent of the dimension. However, there are many situations when AD and the adjoint-state method are inefficient or not applicable. Finding the adjoint equation requires a careful derivation (which depends on the PDE as well as on initial and boundary conditions), and then a numerical method must be implemented to solve it, which takes considerable development time. For this reason complicated codes that are often updated with new features, such as weather models, rarely have the capability to compute a full gradient. There are software packages that solve for the adjoint automatically, or run AD, but these require a programming environment that restricts the user, and may not be efficient in parallel high-performance computing environments.

There is a plethora of derivative-free optimization (DFO) algorithms, including grid search, Nelder–Mead, (quasi-) Monte-Carlo sampling, simulated annealing and MCMC methods [43]. Modern algorithms include randomized methods, Evolution Strategies (ES) such as CMA-ES [36], Hit-and-Run [8] and random cutting planes [17]. Textbook DFO methods ([15, Algo. 10.3], [55, Algo. 9.1]) are based on interpolation and trust-regions. A limitation of all these methods is that they do not scale well to high-dimensions (beyond $\mathcal{O}(10^2)$).

Stochastic gradient-free methods Our stochastic subspace descent (SSD) method (2) has been previously explored under the names "random gradient," "random pursuit," "directional search", and "random search". The algorithm dates back to the 1970s, with some analysis (cf. [24, Ch. 6] and [27, 64]), but it never achieved prominence because zeroth-order methods are not competitive with first-order methods when the gradient is available. Most analysis has focused on the specific case $\ell = 1$ [44, 53, 65]. More recently, the random gradient method has seen renewed interest. For example, [44] analyzes the case when f is quadratic, and [65] provides an analysis (assuming a line search oracle). The method of Gaussian smoothing introduced in [53] is similar to what we propose. We compare the analysis and performance of [53] to that of our method in Sects. 2 and 3. Gaussian smoothing convolves the objective function with a Gaussian random variable to make the objective differentiable without changing its stationary points.

$$f^h(\mathbf{x}) = \mathbb{E}_{\mathbf{u}} f(\mathbf{x} + \mathbf{u}h), \tag{4}$$

for $h \geq 0$ and $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_d)$. It is common (e.g., [5, 6]), and simpler, to consider the case $\boldsymbol{\Sigma}_d = \mathbf{I}_d$. It is shown in [53] that (4) leads to the following finite-difference approximation of the gradient,

$$\nabla f(\mathbf{x}) \approx \nabla f^h(\mathbf{x}) = \mathbb{E}_{\mathbf{u}} \left[\mathbf{u} \frac{f(\mathbf{x} + \mathbf{u}h) - f(\mathbf{x})}{h} \right]. \tag{5}$$

The obvious way to estimate $\nabla f^h(\mathbf{x})$ is the single-sample unbiased estimator proposed by Nesterov,

$$\nabla f^h(\mathbf{x}) \approx \mathbf{u} \frac{f(\mathbf{x} + \mathbf{u}h) - f(\mathbf{x})}{h}.$$

Naturally, such an estimator may have a large variance. Thus, to reduce the variance it is tempting to consider taking $\ell > 1$ and averaging the results as follows

$$\frac{1}{\ell}\sum_{i=1}^{\ell} \mathbf{u}_i \frac{f(\mathbf{x}+\mathbf{u}_i h) - f(\mathbf{x})}{h}, \tag{6}$$

where $\mathbf{u}_1, \dots, \mathbf{u}_\ell \overset{iid}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, as in, e.g., [6]. While independent directional derivatives provide an estimate of the gradient with a reduced variance compared to Gaussian smoothing, independence comes with the undesirable property highlighted in [6]: even $\ell > d$ directional derivatives are insufficient to recover the exact gradient. In this paper we consider an alternative to (6) for approximation of the gradient when $\ell \geq 1$ and $h \to 0$, which is valid when using a derivative oracle such as forward-mode automatic differentiation. Rather than independent Gaussian vectors, we require the $\mathbf{u}_i$ to be orthonormal; Eq. (10) provides a method for generating such vectors. This is discussed further in Sect. 2.2. The use of orthonormal $\mathbf{u}_i$ enables the use of machinery that provides sharper and simpler analysis than previously available. Various proximal, acceleration and noise-tolerant extensions and analyses of Gaussian smoothing have appeared in [6, 22, 23, 29]. Another variant of random gradient has recently been proposed in the reinforcement learning community. The Google Brain Robotics team sampled orthogonal directions to train reinforcement learning systems [14] but treated it as a heuristic to approximate Gaussian smoothing Similarly, [21] uses columns from Haar-distributed matrices and considers the case $\ell = 1$, focusing on technical issues related to the small bias introduced by estimation of directional derivatives by finite differences. The recent papers [6, 13] also investigate techniques similar to ours though like [21] they focus on the implications of the finite difference bias. Following [51] we assume that directional derivatives are available via an oracle such as forward-mode automatic differentiation so the finite-difference bias is of no concern. Analysis using finite-differences in place of exact directional derivatives is possible (see, e.g., [6]). In a forthcoming manuscript we show that in the case $h > 0$, convergence of our algorithm is to within a ball of radius $\mathcal{O}(h^2)$ of the minimum function evaluation where h can be on the order of 10^{-8}; h smaller than 10^{-8} incurs numerical instability errors and should be avoided. For the types of problems discussed in this work, in particular PDE-constrained optimization, precision of this order is often impossible; thus, the error attributable to a biased estimate of the gradient is subsumed by other sources of error such as measurement error or termination of the optimization algorithm prior to convergence. For this reason we omit analysis of the finite-difference case and focus only on the setting of exact directional derivatives.

Alternatives As a baseline one could use $\mathcal{O}(d)$ function evaluations to obtain $\nabla f(\mathbf{x})$ using forward-mode automatic differentiation, which is too costly when d is large and evaluating $f(\mathbf{x})$ is expensive. Once $\nabla f(\mathbf{x})$ is computed, one can run gradient descent, accelerated variants [50], non-linear conjugate gradient methods [35], or quasi-Newton methods like BFGS and its limited-memory variant [55]. In the numerical results section we compare to (finite-difference versions of) gradient descent and BFGS because they are so ubiquitous. We also provide comparisons to

Gaussian smoothing and to coordinate descent as the method we propose generalizes both concepts.

1.2 Structure of this document and contributions

In Sect. 2.1 we investigate convergence of the stochastic subspace descent method for smooth functions. Assumptions used throughout the document are listed, and expected rates of convergence are provided in the case of non-convex, convex, and strongly-convex functions, as well as functions satisfying the Polyak-Lojasiewicz inequality. In Sect. 2.2 we discuss the properties of gradient approximation along random orthogonal directions for use with (2). Choosing directions from a specific distribution that we specify, we are able to provide non-asymptotic, high-probability convergence results for strongly-convex functions. As previously mentioned this algorithm is a generalization of several classical algorithms for which convergence has already been studied, however as stochastic subspace descent has not been previously introduced, the main results are original. In particular, Theorem 1 and Corollary 1 provide a generalization and different analysis than has previously been performed on Gaussian/spherical smoothing. Theorem 2 is a straightforward generalization of a previously known result. Theorem 3 is a generalization of known analysis for convergence of gradient descent on non-convex objectives. Theorem 4 is new analysis. Lemma 1 is probably known but we have not seen it stated as such, and the remarks are known or simple to prove.

In Sect. 3.1 we provide empirical results on a simulated function that Nesterov dubs "the worst function in the world" [52]. In Sect. 3.2 the placement of inducing points for sparse Gaussian processes in the framework of [66] is optimized. As a final empirical demonstration, in Sect. 3.3 our algorithms are tested in the PDE-constrained optimization setting on a shape optimization problem. For the sake of readability, proofs are relegated to the Appendix.

In this document, uppercase boldfaced letters represent matrices, lowercase boldfaced letters are vectors. The vector norm is assumed to be the Euclidean 2-norm, and the matrix norm is the operator norm.

2 Main results

For the remainder of this section we make use of the following assumptions on the sequence of matrices $(\mathbf{P}_k)$ and the function f to be optimized.

Assumption 1 Let $\ell \leq d$ and assume:

(A0) $\mathbf{P}_k \in \mathbb{R}^{d\times\ell}$, $k = 1, 2, \ldots$, are iid random matrices such that $\mathbb{E}\, \mathbf{P}_k\mathbf{P}_k^\top = \mathbf{I}_d$ and $\mathbf{P}_k^\top\mathbf{P}_k = (d/\ell)\, \mathbf{I}_\ell$.

(A1) $f : \mathbb{R}^d \to \mathbb{R}$ is continuously-differentiable with a λ-Lipschitz first derivative.

(A2) The function f attains its minimum f_*.

(A3) For some $0 < \gamma \leq \lambda$ (where λ is the Lipschitz constant in (A1)) and all $\mathbf{x} \in \mathbb{R}^d$, the function f satisfies the Polyak-Lojasiewicz (PL) inequality:

$$f(\mathbf{x}) - f_* \leq \|\nabla f(\mathbf{x})\|^2/(2\gamma). \tag{7}$$

(A3') f is γ-strongly-convex for some $\gamma > 0$ and all $\mathbf{x} \in \mathbb{R}^d$. Note, $\lambda \geq \gamma$ where λ is the Lipschitz constant in (A1).

(A3") f is convex and attains its minimum f_* on a domain $\mathcal{D}$, and there is an $R > 0$ such that for the parameter initialization $\mathbf{x}_0$, $\max_{\mathbf{x},\ \mathbf{x}_* \in \mathcal{D}}\{\|\mathbf{x} - \mathbf{x}_*\| \ : f(\mathbf{x}) \leq f(\mathbf{x}_0)\} \leq R$.

The assumptions on the matrix $\mathbf{P}_k$ can be satisfied by sampling $\ell \leq d$ columns without replacement from an orthogonal matrix. Coercivity of f implies the existence of the constant R in (A3"). For the results below, particularly the rate in Theorem 2, we require knowledge of the value of R. Also note that (A3') implies (A3).

2.1 Asymptotic results

We now provide conditions under which function evaluations $f(\mathbf{x}_k)$ of stochastic subspace descent converge to a function evaluation at the optimum $f(\mathbf{x}_*)$. In the case of a unique optimum we also provide conditions for the iterates $\mathbf{x}_k$ to converge to the optimum $\mathbf{x}_*$. Stochastic subspace descent, so-called because at each iteration the method descends in a random low-dimensional subspace, is a gradient-free method as it only requires computation of directional derivatives at each iteration without requiring direct access to the gradient. In practice we use ℓ columns from a scaled Haar-distributed random matrices to define randomized directions along which to descend at each iteration. However, neither Theorem 1, nor the subsequent theorems in this subsection require Haar-distributed matrices specifically, as long as the random matrices satisfy Assumption (A0). Section 2.2 demonstrates the advantages of using Haar over random coordinate descent type schemes.

Theorem 1 (Convergence of SSD) *Assume (A0), (A1), (A2), (A3) and let* $\mathbf{x}_0$ *be an arbitrary initialization. Then recursion* (2) *with* $0 < \alpha < 2\ell/(d\lambda)$ *results in* $f(\mathbf{x}_k) \xrightarrow{a.s.} f_*$ *and* $f(\mathbf{x}_k) \xrightarrow{L^1} f_*$.

Theorem 1 guarantees L^1 and almost-sure convergence of the function values to a minimizer of the function whenever the function is continuously differentiable, has Lipschitz gradient, and satisfies the PL inequality (A3). A broadly useful example of an objective function satisfying (A3) is linear least squares with a data matrix that is not full column rank; Theorem 1 provides a convergence result for this rank-deficient linear least squares, and similarly well-behaved non-convex functions. Corollary 1(*ii*) shows that the rate of convergence is linear.

Corollary 1 (Convergence under strong-convexity and rate of convergence)

(i) *Assume (A0), (A1), (A2), (A3') and let* $\mathbf{x}_0$ *be an arbitrary initialization. Then recursion* (2) *with* $0 < \alpha < 2\ell/d\lambda$ *results in* $\mathbf{x}_k \xrightarrow{a.s.} \mathbf{x}_*$ *where* $\mathbf{x}_*$ *is the unique minimizer of* f.

(ii) *Assume (A0), (A1), (A2), and either (A3) or (A3'). Then with* $\alpha = \ell/(d\lambda)$, *the recursion* (2) *attains the following expected rate of convergence*

$$\mathbb{E}f(\mathbf{x}_k) - f_* \le \omega^k (f(\mathbf{x}_0) - f_*), \quad \omega = 1 - \ell\gamma/(d\lambda). \tag{8}$$

With $\ell = d$ we recover a textbook rate of convergence, $\omega = 1 - \gamma/\lambda$, for gradient descent [11, §9.3] because, importantly, with $\ell = d$, $\mathbf{P}\mathbf{P}^\top \nabla f(\mathbf{x}) = \nabla f(\mathbf{x})$. This rate is nearly optimal as can be shown with a simple example: let $d = 2$ and $\mathbf{x} = (x, y)$ with initial conditions $\mathbf{x}_0 = (0, 1)$ and $f(\mathbf{x}) = \lambda/2x^2 + \gamma/2y^2$, then $f(\mathbf{x}_k) \to 0$ with linear rate $\omega = (1 - \gamma/\lambda)^2$. Similar results to Corollary 1(*ii*) have been derived for general stochastic gradient methods using techniques described in [10, §4]. Adapting our special case to the general framework of [10] results in the same rate of convergence as corollary 1(*ii*); however [10] does not address different modes of convergence, nor convergence of the iterates. Using the more restrictive assumption of strong-convexity the result of Corollary 1 is much stronger than Theorem 1; we get almost sure convergence of the function evaluations and of the iterates to the optimal solution at a linear rate. In inverse problems the convergence of $\mathbf{x}_k$, rather than that of $f(\mathbf{x}_k)$ is of paramount importance. Furthermore, if either assumption (A3) or (A3') is satisfied, SSD has a linear rate of convergence. The rate of convergence is strictly better than that presented in [53, Thm. 8]. The rate in [53] for γ-strongly convex objectives with λ-Lipschitz gradient is

$$\mathbb{E}f(\mathbf{x}_k) - f_* \le (\lambda/2)(1 - \gamma/(8\lambda(d+4)))^k \|\mathbf{x}_0 - \mathbf{x}_*\|^2. \tag{9}$$

By λ-Lipschitz gradient our Corollary 1 (*ii*) implies

$$\mathbb{E}f(\mathbf{x}_k) - f_* \le (\lambda/2)(1 - \ell\gamma/(d\lambda))^k \|\mathbf{x}_0 - \mathbf{x}_*\|^2,$$

which is strictly better than (9). Note that $\ell = 1$ in (9), while in our case ℓ can be chosen to be greater than one.

The proof in the convex case is different, but substantively similar to a proof of coordinate descent on convex functions found in [71].

Theorem 2 (Convergence under convexity) *Assume (A0), (A1), (A2), (A3"). Then recursion* (2) *with* $\alpha = \ell/(d\lambda)$ *gives*

$$\mathbb{E}f(\mathbf{x}_k) - f_* \le 2d\lambda R^2/(k\ell).$$

Convergence in the convex case is in expectation, and is sub-linear. This is in line with the convergence rate of gradient descent which is also sub-linear in the smooth, convex case [52]. In particular, taking $\ell = d$, our result gives $f(\mathbf{x}_k) - f_* \le 2\lambda R^2/k$ (this is now a deterministic result), where the stepsize is $\alpha = 1/\lambda$. It can be shown that for this common choice of a stepsize, there is a function f satisfying the assumptions of Thm. 2 where $f(\mathbf{x}_k) - f_* \geq \frac{\lambda}{4k+2} R^2$ [20, Thm. 3.2], which implies that when $\ell = d$, the upper bound in Thm. 2 is tight to within a factor of 8.

In the general non-convex setting we can provide guarantees of convergence to a stationary point and are able to provide guarantees on the rate at which $\|\nabla f(\mathbf{x}_k)\|$ decreases. These are presented in the following theorem which adapts well-known results for the convergence of gradient descent on non-convex functions to our case. The rates of convergence are of the same order as [53, p.24] with slightly better constants.

Theorem 3 (Non-convex convergence) *Assume (A0), (A1), (A2). Then recursion* (2) *with $\alpha = \ell/(d\lambda)$ and an arbitrary initialization yields*

$$\min_{i \in \{0,\ldots,k\}} \mathbb{E}\|\nabla f(\mathbf{x}_i)\|^2 \leq \frac{2d\lambda(f(\mathbf{x}_0) - f_*)}{(k+1)\ell}.$$

That is, $k = \mathcal{O}(d/(\ell\epsilon))$ iterations are required to achieve $\mathbb{E}\|\nabla f(\mathbf{x}_k)\|^2 < \epsilon$.

2.2 High-probability results

While it is important to understand how an algorithm will perform on average, in practice it is good know how it is likely to perform on a single run. In this section we discuss convergence bounds that hold with high probability, providing a better understanding of typical convergence. We consider two types of random matrices from the class satisfying assumption (A0):

1. *Columns from Haar-distributed random orthogonal matrix*:

$$\mathbf{P} = \sqrt{d/\ell}\,\mathbf{Q}\mathbf{I}_{d\times\ell} \in \mathbb{R}^{d\times\ell}, \tag{10}$$

   where $\mathbf{Q}$ is as in the QR-decomposition of a matrix $\mathbf{Z} = \mathbf{Q}\mathbf{R} \in \mathbb{R}^{d\times d}$ with $\mathbf{R}_{ii} > 0$, and each element of $\mathbf{Z}$ is drawn independently from $\mathcal{N}(0,1)$. $\mathbf{I}_{d\times\ell}$ truncates $\mathbf{Q}$ to its first ℓ columns so $\mathbf{Q}\mathbf{I}_{d\times\ell}$ corresponds to ℓ columns of the random orthogonal matrix distributed according to the Haar measure on orthogonal matrices [49]. In fact, for our results to hold, $\mathbf{R}_{ii}$ need not be strictly positive, we merely require that $\mathbf{P}\mathbf{P}^\top \nabla f(\mathbf{x}) \overset{d}{=} (d/\ell)\mathrm{Proj}_{\mathrm{col}(\mathbf{Z}\mathbf{I}_{d\times\ell})}(\nabla f(\mathbf{x}))$. It is convenient to work with Haar distributed matrices so we use matrices of the form (10).

   There is an important correspondence between matrices described by (10), Gaussian smoothing of [53], and the smoothing on a sphere of [6]. Let $\mathbf{Q} \in \mathbb{R}^{d\times d}$ be as in (10), $\mathbf{v} \in \mathbb{R}^d$ an arbitrary fixed vector, and $\mathbf{u} = \mathbf{Z}\mathbf{e}_1$, where $\mathbf{e}_1$ is the first standard basis vector, so $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$. Then $\mathbf{Q}^\top\mathbf{v}/\|\mathbf{v}\|$ and $\mathbf{u}/\|\mathbf{u}\|$ are both distributed uniformly on the d-dimensional sphere. When $\ell = 1$, $\mathbf{P} = \sqrt{d}\,\mathbf{Q}\mathbf{e}_1 \overset{d}{=} \sqrt{d}\,\mathbf{u}/\|\mathbf{u}\|$. Therefore, in this case $(\|\mathbf{u}\|^2/d)\mathbf{P}\mathbf{P}^\top \overset{d}{=} \mathbf{u}\mathbf{u}^\top$. That is, when $h = 0$ our method is proportional to Gaussian smoothing with a constant of proportionality $\|\mathbf{u}\|^2/d$. Since $\|\mathbf{u}\|^2$ is a χ^2 random variable with mean d, this

Table 1 Summary of special cases of our framework

Description	ℓ	$\mathbf{P}_k$	α
Gaussian smoothing	1	Satisfying (10)	$\lVert \mathbf{Z}_k \mathbf{e}_1 \rVert^2/(d^2\lambda)$
Smoothing on a sphere of radius $\sqrt{d}$	1	Satisfying (10)	$0 < \alpha < 2/(d\lambda)$
Gradient descent	d	Any satisfying (A0)	$0 < \alpha < 2/\lambda$
Block-coordinate descent	$1 \le \ell < d$	Satisfying (11)	$0 < \alpha < 2\ell/(d\lambda)$

Using the ℓ, $\mathbf{P}_k$, and α specified in the table it is possible to recover exactly the methods described. $\mathbf{Z}_k$ is the kth Gaussian matrix used to generate $\mathbf{P}_k$ and $\mathbf{e}_1$ is the first standard basis vector

means in high dimensions the constant of proportionality is sharply concentrated around 1.

Furthermore, when $\ell = 1$, then $\mathbf{P} \sim \mathcal{U}(S(0, \sqrt{d}))$, i.e., we recover spherical smoothing as discussed by [6]. To increase ℓ, the traditional method in the literature is to use (6), which is different than (10) for $\ell > 1$. Thus, we provide a novel generalization of Gaussian smoothing and smoothing on the sphere as a mapping of the gradient onto a lower dimensional subspace. A consequence of our approach is that matrices of the form (10) with $\ell = d$ satisfy $\mathbf{P}\mathbf{P}^\top = \mathbf{P}^\top\mathbf{P} = \mathbf{I}_d$, and the exact gradient is recovered.

To summarize, in high-dimensions, our method with matrices defined by (10) is very similar to both Gaussian smoothing and smoothing on a sphere for $\ell = 1$, but the differences with existing methods grow as ℓ increases, and are markedly different for $\ell = d$. Table 1 provides a summary of the well-known special cases to our algorithm, and describes how they relate to our framework. We re-emphasize that for both Gaussian smoothing and smoothing on a sphere it is typical to use $h > 0$, however we only analyze the case $h = 0$. Indeed it is true the $h = 0$ is no longer smoothing of the gradient *per se*, but is a projection of the gradient onto a subspace of dimension ℓ.

Note that for problems of interest, function evaluations are so costly that we can ignore the computational overhead of a QR decomposition, which is $\mathcal{O}(d\ell^2 - 2\ell^3/3)$. Since $\ell \ll d$, the cost is negligible compared to, for instance, d PDE-solves.

2. *Randomized block-coordinate descent random matrix*:

$$\mathbf{P} = \sqrt{d/\ell}\,\mathbf{D}, \tag{11}$$

where $\mathbf{D} \in \mathbb{R}^{d\times\ell}$ is comprised of ℓ columns of the identity matrix $\mathbf{I}_d$ selected uniformly at random. It is straightforward to verify that (10) and (11) satisfy assumption (A0), the former by properties of the QR decomposition.

Denoting the columns of $\mathbf{P}$ as $\mathbf{P}^1, \dots, \mathbf{P}^\ell$, the following equality holds

$$\nabla f(\mathbf{x}) \approx \mathbf{P}\mathbf{P}^\top \nabla f(\mathbf{x}) = \begin{pmatrix} \mathbf{P}\nabla_{\mathbf{P}^1} f(\mathbf{x}) \\ \vdots \\ \mathbf{P}\nabla_{\mathbf{P}^\ell} f(\mathbf{x}) \end{pmatrix}, \tag{12}$$

where $\nabla_{\mathbf{P}^i} f(\mathbf{x})$ is a directional derivative of f at $\mathbf{x}$ in the $\mathbf{P}^i$-th direction. Thus there is a convenient interpretation that the gradient is approximated by a mapping onto an ℓ-dimensional random subspace embedded in $\mathbb{R}^d$. In fact, since $\mathbb{E}\mathbf{P}\mathbf{P}^\top = \mathbf{I}$, $\mathbf{P}\mathbf{P}^\top \nabla f(\mathbf{x})$ is centered at $\nabla f(\mathbf{x})$ with MSE $(1 - \ell/d)\|\nabla f(\mathbf{x})\|^2$.

In advance of the main results of this section we investigate how well multiplication by the matrices specified by (10) and (11) preserves the norm of an arbitrary vector. Norm invariance has important consequences with respect to the rate of convergence. Of particular interest for our purpose is the lower bound which governs the rate of convergence (see Theorem 4 for details). We define a successful embedding in order to quantify the norm invariance.

Definition 1 (Successful isometric embedding) An embedding $\mathbf{P}$ is deemed to be successful if for some $\epsilon \in (0,1)$ and some $\mathbf{v} \in \mathbb{R}^d$, $\|\mathbf{P}^\top \mathbf{v}\|^2 \geq (1-\epsilon)\|\mathbf{v}\|^2$.

The following Lemma provides the probability of successful embedding when the matrix $\mathbf{P}$ is Haar-distributed.

Lemma 1 (Approximately isometric embedding using Haar-distributed matrices) *Fix* $\epsilon \in (0,1)$, *a positive integer* $\ell \leq d$, *and consider a matrix* $\mathbf{P}$ *drawn according to* (10). *Then for any fixed vector* $\mathbf{v} \in \mathbb{R}^d$, *the probability of a successful embedding,* δ, is given by

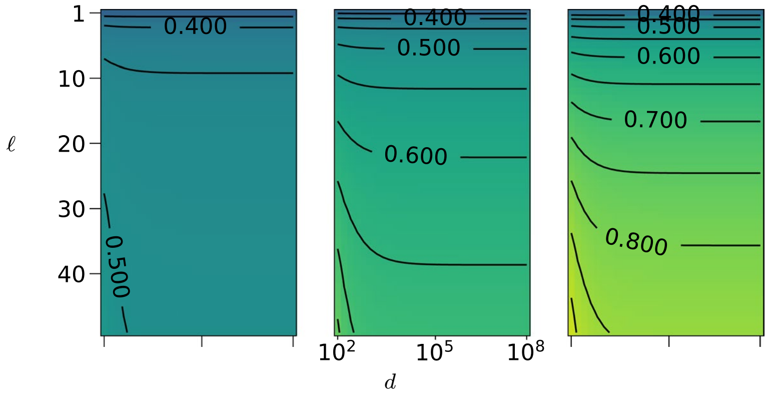

Fig. 1 Contour plots for probability of successful embedding for various values of ℓ, d, and ϵ. Each of the figures share the same horizontal and vertical range. **Left**: $\epsilon = 0.01$. **Center**: $\epsilon = 0.1$. **Right**: $\epsilon = 0.2$

$$\delta = 1 - I_{(1-\epsilon)\ell/d}(\ell/2, (d-\ell)/2) = \mathbb{P}(X \geq (1-\epsilon)\ell/d),$$

where $I_p(\alpha, \beta)$ *is the regularized incomplete Beta function, and* $X \sim \mathcal{B}eta(\ell/2, (d-\ell)/2)$.

For fixed d one can simply use Lemma 1 to determine values of ℓ and ϵ required to achieve the desired probability of successful embedding. For a fixed d, an increase in ℓ or ϵ corresponds to an increase in δ. For $\ell = d$, we can take $\epsilon = 0$ and $I_1(\alpha, \beta) = 0$ so $\delta = 1$, meaning we always have a perfect isometric embedding. Figure 1 provides examples of the probability of success for various values of ℓ, d, and ϵ. It is plain to see the similarity between the left hand-side of Lemma 1 and the lower tail of the Johnson–Lindenstrauss (JL) lemma when it is applied to a single point. Indeed, a connection of the JL lemma with the Beta distribution is discussed in [26]. Our bound differs in two ways: first, in [26] they provide asymptotic results as $d \to \infty$ whereas our results are valid for all d with the d-dependence explicit; second, [26] provide a closed-form approximate bound while we provide an exact functional form. For finite dimensions, our result is stronger.

A well-known property of the matrices (10) is that $\mathbf{P}^\top$ is spherically symmetric. That is $\mathbf{P}^\top\mathbf{u}$ has the same distribution as $\mathbf{P}^\top$ for any orthogonal matrix $\mathbf{u}$. Consequently, the quality of the embedding does not depend on the vector $\mathbf{v} \in \mathbb{R}^d$. Naturally, the coordinate descent matrices given by (11) do not share this orthogonal invariance; indeed, speaking of the ability of such matrices to preserve pairwise distances, Achlioptas [2] says "A naive, perhaps, attempt at constructing JL-embeddings would be to pick ℓ of the original coordinates in d-dimensional space as the new coordinates. Naturally, as two points can be very far apart while only differing along a single dimension, this approach is doomed". Remark 1 provides intuition for the reason randomized block-coordinate descent cannot be close to norm preserving for arbitrary directions.

Remark 1 (Coordinate sampling is rarely an isometry) Let $\mathbf{v} \in \mathbb{R}^d$ be a standard basis vector and $\mathbf{P} \in \mathbb{R}^{d\times\ell}$ be a coordinate descent sampling matrix satisfying (11). Then, $\|\mathbf{P}^\top\mathbf{v}\| \in \{0, 1\}$, and

$$\mathbb{P}(\|\sqrt{\ell/d}\,\mathbf{P}^\top\mathbf{v}\|^2 = 1) = \ell/d \quad \text{and} \quad \mathbb{P}(\|\sqrt{\ell/d}\,\mathbf{P}^\top\mathbf{v}\|^2 = 0) = 1 - \ell/d.$$

Thus,

$$\mathbb{E}\|\mathbf{P}^\top\mathbf{v}\|^2 = 1 \quad \text{and} \quad \mathbb{V}\mathrm{ar}\|\mathbf{P}^\top\mathbf{v}\|^2 = d/\ell - 1.$$

Since exactly ℓ entries of $\mathbf{P}\mathbf{P}^\top$ are 1, the probability that any non-zero entry corresponds to a non-zero entry of $\mathbf{v}$ is ℓ/d.

Remark 1 shows that in the worst case (that is, if the vector $\mathbf{v}$ is axis-aligned with concentration along a single coordinate), there is no approximate norm-preservation: it is either exact with probability ℓ/d or not-at-all with probability $1 - \ell/d$. This compares very unfavorably to the results of Lemma 1, cf. Fig. 1.

To summarize, the structure of the objective function plays a role in the quality of a coordinate descent mapping, and in the worst-case the mapping using (11) is useless with probability $1-\ell/d$. In contrast, using Haar matrices guarantees that irrespective of the structure of the function a successful embedding is obtained with probability according to Lemma 1. Though this probability depends on dimension, it is not very sensitive to an increase in d, as illustrated in Fig. 1.

Due to the strong dependence on the dimension for randomized coordinate descent, the analysis in the remainder of this section is not appropriate for matrices of type (11). Thus, we consider only Haar distributed random matrices. It should be noted that there are special classes of functions for which the complexity is independent of d, as discussed in [39], however in general the dependence on the dimension can not be removed using coordinate descent methods. We consider first a result that is a simple but useful corollary to Theorem 3.1 in [48], later proved in [4]

Remark 2 Let $B \sim Bin(k,\delta)$. Then for all $t>0$ and $\delta \in (0,1)$

$$\mathbb{P}(B > k\delta + t) \le \exp\left(-t^2/(2\sigma_k^2)\right) \quad \text{and} \quad \mathbb{P}(B < k\delta - t) \le \exp\left(-t^2/(2\sigma_k^2)\right)$$

with

$$\sigma_k^2 = \begin{cases} \frac{k(1-2\delta)}{2\log((1-\delta)/\delta)} & \delta \in (0,1)\setminus\{1/2\} \\ 1/4, & \delta = 1/2. \end{cases} \tag{13}$$

Remark 2 provides an optimal proxy-variance for sub-Gaussianity of Binomial random variables. For $\delta = 1/2$, σ_k^2 is defined as $k/4$ so that σ_k is continuous in δ; also note that for any k, $\lim_{\delta\nearrow 1}\sigma_k^2 = 0$ which agrees with the fact $\mathbb{P}(B \ne k) = 0$ in the case $\delta = 1$ (which occurs when $\ell = d$). We use the result of Remark 2 to provide sharp bounds for the performance of our algorithm. First we state a result showing that the success of each embedding is independent so that the number of successful embeddings can be treated as a binomial random variable, which in turn allows for an application of Remark 2.

Remark 3 Let $A_k(\mathbf{v}_k) = \left\{\left\|\mathbf{P}_k^\top \mathbf{v}_k\right\|^2 \le (1-\epsilon)\|\mathbf{v}_k\|^2\right\}$ and $\mathbf{v}_k$ be independent of $\mathbf{P}_k$ for all k with $\mathbf{P}_k$ drawn according to (10). Then $(A_k(\mathbf{v}_k))$ is an independent sequence of events.

The remark is proved by iteratively conditioning on the available information and recognizing that spherical symmetry implies A_k is identically distributed for any $\mathbf{v}_k$ that is fixed or independent of $\mathbf{P}_k$, and can be found in the Appendix. Using Lemma 1 and Remark 2 in conjunction with Remark 3 results in the following probabilistic rate of convergence,

Theorem 4 (Probabilistic rate of convergence. Strongly-convex case) *Assume (A1), (A2), (A3') and let* $\mathbf{x}_0$ *be an arbitrary initialization. Apply recursion* (2) *with* *step-size* $\alpha = \ell/(d\lambda)$ *and* $\mathbf{P}_k$ *drawn according to* (10), *with* ℓ *sufficiently large to achieve the desired* ϵ *and* δ *according to Lemma* 1. *Then for any* $t \in (0, \delta]$

$$\mathbb{P}\big(f_e(\mathbf{x}_k) \geq \rho^k f_e(\mathbf{x}_0)\big) \leq \exp(-(kt)^2/2\sigma_k^2),$$

where σ_k^2 *is defined by* (13) *and,*

$$\rho = \left(1 - (1-\epsilon)\frac{\ell\gamma}{d\lambda}\right)^{\delta - t}.$$

Theorem 4 provides an exponential decay (in k) for the probability that any single run of the algorithm converges more slowly than the average performance guaranteed by Corollary 1(*ii*). Similar results can be derived for the convex case combining the methodology of Theorem 4 with Theorem 2. We note the interplay between parameters δ, t and ϵ, all of which affect ρ: we can trade off $\epsilon \to 0$ by decreasing δ; both ϵ and δ affect ρ, but due to their complicated relationship, it is not easy to optimize ρ with respect to these parameters. We can also take $t \to \delta$ to get a conservative bound (high probability but worse rate ρ), or $t \to 0$ to get an aggressive bound (lower probability but better rate ρ). Since the probability concentrates quickly with the number of iterations k, for large iterations, one can take $t \propto 1/\sqrt{k}$ and have both good control over the failure probability, $\exp(O(-k))$, while still having a good convergence rate.

Again, Theorem 4 agrees with the standard deterministic result (discussed after Corollary 1) when $\ell = d$, since then $\epsilon = 0$, $\delta = 1$ and $\sigma_k^2 = 0$ for any $t > 0$, so the rate ρ is arbitrarily close to $1 - \frac{\gamma}{\lambda}$, which is the rate from Corollary 1(*ii*).

3 Experimental results

In this section we provide results for a synthetic problem, a problem from the machine learning literature, and a PDE-constrained shape-optimization problem.

In the synthetic and machine learning problems we compare to randomized block-coordinate descent. For the shape-optimization problem we compare to Gaussian smoothing and finite-difference gradient descent. In each of the examples we make use of a deterministic backtracking line search with Armijo conditions. Analysis of algorithms with inexact gradient estimates using a stochastic line search is a topic that has received considerable attention recently, but which we do not address here. In particular, [7, 56] describe a stochastic variant of an Armijo backtracking line search that can be adapted to our method to provide sharper convergence analysis. Their work does not directly apply to all of our settings without modification, but in the strongly-convex case the application is clear.

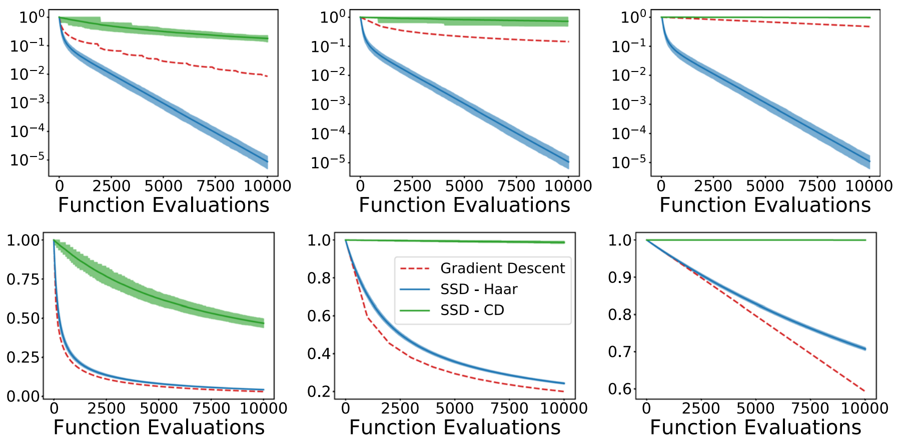

Fig. 2 Minimizing a function from the family (14) with $r = 20,\ \lambda = 8$. CD represents randomized block-coordinate descent. In several of the subfigures gradient descent overlaps randomized block-coordinate descent. The shaded regions in the SSD cases represent the interval between best 10th and 90th percentile performance after 1000 runs. The vertical-axis is the relative error: $(f(\mathbf{x}_k) - f_*)/f_*$. **Left**: $d = 100$. **Center**: $d = 1000$. **Right**: $d = 10{,}000$. **Top**: Step-size chosen by a backtracking linesearch with Armijo conditions. **Bottom**: Fixed step-size

3.1 Synthetic data

We begin with a simulated example using what Nesterov dubs "the worst function in the world" [52]. Fix a Lipschitz constant $\lambda > 0$ and let

$$f_{\lambda,r}(\mathbf{x}) = \lambda((x_1^2 + \sum_{i=1}^{r-1} \left(x_i - x_{i+1}\right)^2 + x_r^2)/2 - x_1)/4, \tag{14}$$

where x_i represents the i^{th} coordinate of $\mathbf{x}$ and $r < d$ is a constant integer that defines the intrinsic dimension of the problem. This function is convex and continuously differentiable with global minimum $f_* = -\lambda r/8(r+1)$, so Theorem 2 applies. This example illustrates the consequences of the dimension dependence in Remark 1, as well as the dimension independence of Lemma 1 in the context of optimization using recursion (2). Figure 2 highlights the performance of three algorithms: finite-difference gradient descent, SSD using (11) (hereafter, SSD-CD), and SSD using (10) (hereafter, SSD-Haar); all algorithms start at $\mathbf{x} = 0$. We show each with the fixed step-size $\alpha = \ell/(d\lambda)$ suggested by the theorem, as well as an adaptive step-size using a backtracking linesearch with the Armijo conditions. We keep $\ell = 3$ and $r = 20$ fixed and provide results for $d = 100$, $d = 1000$, $d = 10{,}000$. For the SSD cases we run each 500 times and display the performance of the 10^{th} and 90^{th} percentile (shaded region) as well as the mean performance.

Clearly both gradient descent and randomized block-coordinate descent depend strongly on the ambient dimension of the problem, even when a linesearch is used. Functions from this family are a worst case for both of these algorithms as only the first r dimensions have a non-zero gradient. Thus, in the case $d = 10{,}000$, gradient descent must perform 10,000 function evaluations at every iteration when only $r = 20$ dimensions are important. Similarly, randomized coordinate descent has only a 20/10,000 chance of descending at all, so as predicted in the discussion of Remark 1, we see many iterations of coordinate descent with no improvement. The linesearch makes coordinate descent slower relative to gradient descent for this example because every iteration for which a pertinent coordinate is not selected requires several function evaluations to perform the linesearch. Regarding SSD-Haar, using a linesearch dramatically impacts performance by allowing for invariance to ambient dimension as suggested by Lemma 1. Without linesearch, as expected by Theorem 2, the performance can be no better than that of gradient descent. As previously noted, the function has low intrinsic dimension; the performance on this problem suggests that the bound in Lemma 1 (and in turn, of Theorems 1 and 2) can be sharpened by accounting for this structure and we consider this a promising avenue for future research.

3.2 Parameter estimation for sparse Gaussian processes

We test the efficacy of SSD-Haar against SSD-CD in the context of hyperparameter estimation for sparse Gaussian processes used in regression. The goal is inference on a function $T : \mathbb{R}^d \to \mathbb{R}$ based on noisy observations at m points $\mathbf{z}_1, \ldots, \mathbf{z}_m$. We use a zero-mean Gaussian process with covariance function $\mathbb{C}\mathrm{ov}(T(\mathbf{z}_i), T(\mathbf{z}_j)) = K(\mathbf{z}_i, \mathbf{z}_j; \boldsymbol{\theta})$ and model the m observations as $y_i = T(\mathbf{z}_i) + \epsilon_i$, where $K(\cdot, \cdot; \boldsymbol{\theta})$ is a symmetric positive-definite kernel with parameters $\boldsymbol{\theta}$. The process T is assumed to be independent of the noise vector $(\epsilon_1, \ldots, \epsilon_m)^\top \sim N(\mathbf{0}, \sigma^2\mathbf{I})$ with unknown variance σ^2. We denote the covariance of the vector $\mathbf{T} = (T(\mathbf{z}_1), \ldots, T(\mathbf{z}_m))^\top$ as $\boldsymbol{\Sigma}_{\mathbf{T}} = \mathbb{V}\mathrm{ar}(\mathbf{T})$, where $(\boldsymbol{\Sigma}_{\mathbf{T}})_{ij} = K(\mathbf{z}_i, \mathbf{z}_j; \boldsymbol{\theta})$. Maximum likelihood estimates of the parameters $\boldsymbol{\Theta} = [\boldsymbol{\theta}, \sigma^2]$ are obtained by maximizing the log-marginal likelihood of observations $\mathbf{y} = (y_1, \ldots, y_m)^\top$ with density $p_{\mathbf{y}}$ [69]: $\ell(\boldsymbol{\Theta}; \mathbf{y}) = \log p_{\mathbf{y}}(\mathbf{y}; \boldsymbol{\Theta})$. When the number of observations is large the cost of this maximization is $\mathcal{O}(m^3)$ due to the inversion and determinant calculations in $\ell(\boldsymbol{\Theta}; \mathbf{y})$. We use the method described in [66] to approximate the likelihood. The basic idea is as follows: choose a $p < m$ and define a set of inducing points $\widetilde{\mathbf{z}}_1, \ldots, \widetilde{\mathbf{z}}_p \in \mathbb{R}^d$ different from the original $\mathbf{z}_1, \ldots, \mathbf{z}_m$, and let $\widetilde{\mathbf{T}} = (T(\widetilde{\mathbf{z}}_1), \ldots, T(\widetilde{\mathbf{z}}_p))^\top$. We obtain a lower bound for the loglikelihood [66]:

$$\ell(\boldsymbol{\Theta}; \mathbf{y}) \geq f(\widetilde{\mathbf{z}}_1, \ldots \widetilde{\mathbf{z}}_m, \boldsymbol{\Theta}) = \widetilde{\ell}(\boldsymbol{\Theta}; \mathbf{y}) - \mathrm{tr}(\mathbb{V}\mathrm{ar}(\mathbf{T} \mid \widetilde{\mathbf{T}}))/2\sigma^2. \tag{15}$$

Here $\widetilde{\ell}$ is the loglikelihood of the multivariate Gaussian $N(\mathbf{0}, \widehat{\boldsymbol{\Sigma}}_{\mathbf{T}})$, where $\widehat{\boldsymbol{\Sigma}}_{\mathbf{T}} = \boldsymbol{\Sigma}_{\mathbf{T}} - \mathbb{V}\mathrm{ar}(\mathbf{T} \mid \widetilde{\mathbf{T}}) = \mathbb{C}\mathrm{ov}(\mathbf{T}, \widetilde{\mathbf{T}})\, \boldsymbol{\Sigma}_{\widetilde{\mathbf{T}}}^{-1}\, \mathbb{C}\mathrm{ov}(\widetilde{\mathbf{T}}, \mathbf{T})$ is the the Nyström approximation of $\boldsymbol{\Sigma}_{\mathbf{f}}$ introduced in [70]. Gradient-based methods are used to simultaneously find an optimal placement of the p inducing points and the best hyperparameter settings by maximizing the lower bound in (15), which we re-state as a function of $\mathbf{x} = [\widetilde{\mathbf{z}}_1, \ldots, \widetilde{\mathbf{z}}_p, \boldsymbol{\Theta}]$ to be consistent with notation in previous sections:

$$f(\mathbf{x}) = \widetilde{\ell}(\boldsymbol{\Theta}; \mathbf{y}) - \mathrm{tr}(\mathbb{V}\mathrm{ar}(\mathbf{T} \mid \widetilde{\mathbf{T}}))/2\sigma^2. \tag{16}$$

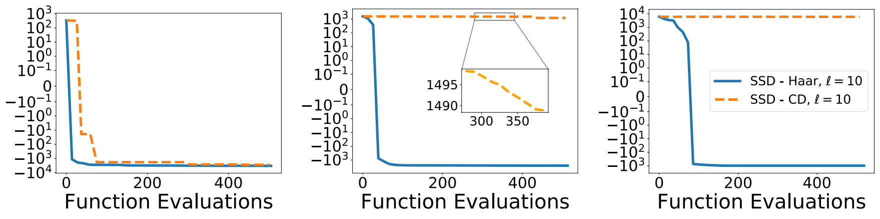

Fig. 3 Approximating a function from the family (14) with a sparse Gaussian process where the hyperparameters have been estimated by minimizing (16) and $r = d$, $\lambda = 1$. CD represents randomized block-coordinate descent. Step-size in all cases is chosen by a backtracking linesearch with Armijo conditions. Left: $d = 3$, $p = 50$, total parameters = 153. Center: $d = 10$, $p = 50$, total parameters = 503. Right: $d = 20$, $p = 100$, total parameters = 2003

Practically speaking, the optimization problem is $(pd + |\theta| + 1)$-dimensional: pd for p inducing points in $\mathbb{R}^d$, $|\theta|$ for the kernel hyperparmeters, and 1 for the unknown noise variance. By moving to this high-dimensional optimization problem the time complexity is reduced to $\mathcal{O}(mp^2)$ and the storage costs to $\mathcal{O}(mp)$.

For example, we model a noisy version of the function described by (14) with $\lambda = 1$ and $r = d$ using a Gaussian process in the framework of [66] with a squared-exponential kernel that has two unknown parameters. Between the inducing points, the parameters of the kernel, and the unknown noise, there are 153, 503, 2003 parameters to be estimated for cases $(d = 3,\ p = 50)$, $(d = 10,\ p = 50)$, $(d = 20,\ p = 100)$ respectively. We report the objective function, which is (16) up to an irrelevant constant. We terminate the algorithm after 500 function evaluations. Thus, since the second and third experiments have 503, and 2003 parameters respectively, gradient descent would not have the opportunity for even one iteration. As such, for all three experiments we only compare SSD-Haar to SSD-CD (Fig. 3).

The objective function of this problem is non-convex despite the underlying function T being convex. The interpretation of coordinate descent is interesting as each coordinate in parameter space either corresponds to one of the hyperparameters of the kernel, to the noise, or to the placement of one of the inducing points along one dimension. Since $r = d$, the latent function has no low-dimensional structure and movements in any direction in input space correspond to a changing function

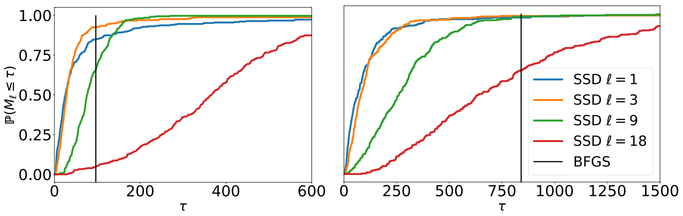

Fig. 4 Left: 30-dimensional problem. Right: 60-dimensional problem. M_ℓ is the number of function evaluations required to attain a cut-off threshold for various values of ℓ. For a fixed initialization BFGS is non-random, represented by the vertical line. Gradient descent, not pictured, has a vertical line at $\tau = 2850$ and $\tau = 22828$ for $p = 30$ and $p = 60$, respectively. $\ell = 1$ is equivalent to the method proposed in [53] when $h = 0$

evaluation. Once again coordinate descent does not scale well with the dimension. This behavior is to be expected: changing the location of particular inducing points along the correct axis has a large improvement on the objective, but if the wrong point is chosen, or the correct point but wrong axis, then little improvement is made (though as we see from the inset, there is slight improvement at each iteration). In contrast, SSD-Haar changes all inducing points in tandem so it descends more rapidly and consistently, particularly in high-dimensional problems. We notice that as before SSD-Haar remains robust to changes in the ambient dimension of the parameter space, though we do see a slight degradation of performance with increased dimension.

We use performance profiles [19] to determine the effect of varying ℓ for different problem sizes and to gauge the variability between runs for a fixed ℓ. A performance profile is conducted by running each parameterization on a suite of randomized restarts, with termination after some pre-specified tolerance for accuracy has been reached. We count the proportion of realizations from each parameterization that achieves the specified tolerance within τ function evaluations where $\tau = 1$ is the fewest function evaluations required in any of the trials, $\tau = 2$ is twice as many function evaluations, etc. Each parameterization is run 300 times. Results for SSD-Haar are shown in Fig. 4 for 30- and 60- dimensional objective functions.

The cut-off threshold is 95% of the distance between the objective function at the parameter initialization and at the optima, as found by BFGS. Clearly, $\ell = 18$ is not a good option in this case. Similarly, $\ell = 9$ can be ruled as it underperforms $\ell = 1$ and $\ell = 3$ approximately 90% (resp. 99%) of the time in the 30- (resp. 60-) dimensional problem. The case $\ell = 1$ has the best single performance: in the fastest trial it is roughly 100 (resp. 800) times faster than BFGS for the 30- (resp. 60-) dimensional problem, but the variance of the performance for $\ell = 1$ is high, and about 1% of the time it performs at least 10 times slower than BFGS (not pictured). On the other hand, $\ell = 3$ beats BFGS by a similar factor and seems to be insulated from the high variance observed for $\ell = 1$. Note also that in 60 dimensions $\ell = 3$ is approximately three times faster than BFGS in 90% of the trials, and about 100 times faster

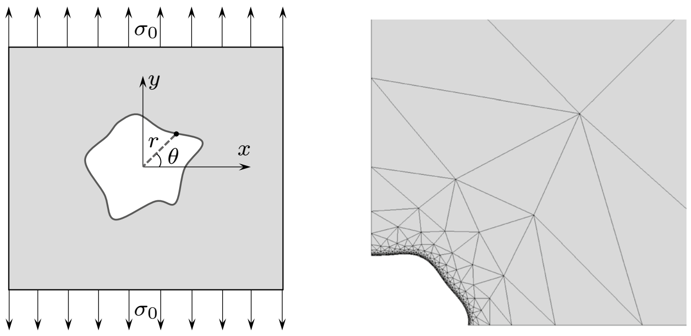

Fig. 5 Left: Schematic of the linear elasticity problem used in the shape optimization example of Sect. 3.3. Right: Conforming finite element mesh used to solve for maximum stress σ_y along the y direction. Only a quarter of the plate corresponding to $\theta \in [0, \pi/2]$ is modeled

in 40% of trials. A few trials of $\ell = 1$ and $\ell = 3$ found their way to a local minima, resulting in the methods not achieving the target threshold.

3.3 Shape optimization

We consider a shape optimization problem involving a linear, elastic structure. Consider a square plate of size 250×250 with a hole, subject to uniform boundary traction σ_0=1, as illustrated in Fig. 5. We adopt a discretize-then-optimize approach to solving the PDE-constrained optimization problem. The discretization and optimization steps do not generally commute and an optimize-then-discretize approach may be preferable for some types of problems [32, §2.9], but we do not pursue this question here.

Our goal is to identify a shape of the hole that minimizes the maximum stress σ_y along the y direction over a quarter of the plate corresponding to $\theta \in [0, \pi/2]$. To this end, we parameterize the radius of the hole for a given θ (see Fig. 5) via

$$r(\theta) = 1 + \delta \sum_{i=1}^{p} i^{-1/2} \left(\xi_i \sin(i\theta) + v_i \cos(i\theta) \right), \tag{17}$$

where $\delta \in (0, 0.5/\sum_{i=1}^{p} i^{-1/2})$ is a user-defined parameter controlling the potential deviation from an n-gon of radius 1. The parameters that dictate the shape are $\boldsymbol{\xi} \in \mathbb{R}^p$ and $\boldsymbol{v} \in \mathbb{R}^p$ so that the parameter space is dimension $d = 2p$. Subscripts indicate the index of the vector. We set $\delta = 0.4/\sum_{i=1}^{p} i^{-1/2}$ so that the minimum possible radius of any particular control point is 0.2 at the initialization. We initialize the entries of $\boldsymbol{\xi}$ and $\boldsymbol{v}$ uniformly at random between -1 and 1. For each instance of $\boldsymbol{\xi}$ and $\boldsymbol{v}$—equivalently $r(\theta)$—we generate a conforming triangular finite element mesh of the plate that we subsequently use within the FEniCS package [46] to solve for the maximum stress σ_y. A mesh refinement study is performed to ensure the spatial discretization errors are negligible. As we only model a quarter of the plate, we apply symmetry boundary conditions so that y and x displacements along $\theta = 0$ and $\theta = \pi/2$ are zero. The Young's modulus and Poisson's ratio of the plate material are set to $E = 1000$ and $\nu = 0.3$, respectively. A similar problem has been examined in [18] using a bi-fidelity variant of the popular SVRG algorithm [41]. Due to the

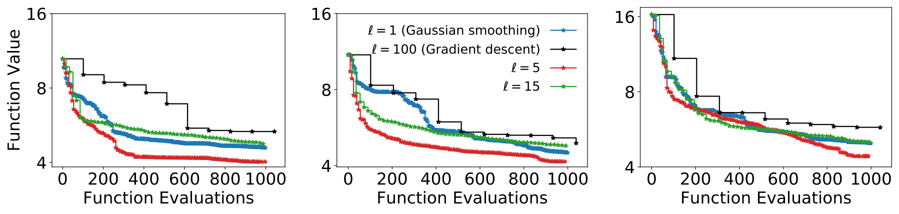

Fig. 6 Three runs for optimization of the objective for a hole with shape parameterized by (17) with $p = 50$ (100 dimensions). Each restart represents an initialization of the parameters uniformly at random in $(-1, 1)$

different focus of that work, the investigation of [18] is conducted in a low-dimensional setting with $d = 6$ rather than $d = 100$ as in our case.

The parametric radius defined by (17) enables us to scale the complexity of the problem arbitrarily by increasing the dimension d. In effect, if d is large then the problem becomes ill-conditioned since ξ_p and ν_p each make at most $\delta p^{-1/2}$ additive contribution to the radius. Such ill-posedness suggests that gradient descent ought to perform poorly as it does not account for the curvature of the objective function. Based upon the intrinsic dimensionality results presented in Sect. 3.1 we anticipate SSD to outperform gradient descent even though it does not explicitly account for the curvature either. Note that each function evaluation requires a PDE-solve meaning that gradient descent requires $d + 1$ PDE-solves per iteration. Though a conforming finite element mesh is used to reduce the computational burden, the cost of so many PDE-solves makes this problem intractable in high-dimensions unless the resolution of the mesh is very low. On the other hand, SSD requires far fewer PDE-solves per iteration provided $\ell \ll d$. As mentioned above, the goal is to minimize the maximum stress in the y-direction, σ_y, over the plate. We make two slight changes to this objective for the sake of the model. First, the stress is obviously minimized if the radius of the hole is zero so we add a term to the objective to penalize deviations from an area of 1 squared unit; even with the regularizer the objective function is still non-convex. Second, the *max* function is not smooth, so it does not fit into the framework of our theory; instead, we minimize the ℓ_p-norm of the stress with $p = 100$, which provides an almost indistinguishable result.

In Fig. 6 we minimize the objective for a hole with shape governed by (17) for problems with $p = 50$ (that is, $d = 100$ parameters), using gradient descent and Gaussian smoothing, as well as SSD with $\ell = 5$ and $\ell = 15$. In each case, an Armijo backtracking linesearch is used.

In all three randomized restarts finite-difference gradient descent performs poorly relative to the stochastic optimizers. The early iterations are particularly good for the stochastic optimizers. We hypothesize that as the ℓ-dimensional subspace along which SSD and Gaussian smoothing descends changes with each iteration, parameter space is explored more thoroughly than deterministic methods, making these subspace methods less likely to get funnelled into long, shallow basins; this is intuitively similar to the recent line of research suggesting that noisy perturbation of iterative algorithms helps avoid saddle points [28]. Alternative perspectives hold that subspace methods are cheap on a per-iteration basis so temporarily being caught in a shallow basin is not as expensive in terms of function evaluations. Conversely, a subspace comprised of a single directional derivative (as in Gaussian smoothing) will have a large variance, causing erratic movements through parameter space whenever the gradient is poorly approximated. Figure 6 corroborates the evidence provided in Fig. 4 that choosing ℓ greather than 1 but less than d can be beneficial in terms of rate of convergence.

It is unclear how to choose an optimal ℓ. Intuition and empirical evidence suggests that a good choice of ℓ depends on all of the eigenvalues of f, not just on the condition number. In particular, we observe that a rapidly decaying eigenspectrum (as in this problem, and to a larger extent the synthetic data problem described in

Sect. 3.1) allows for ℓ to be chosen small compared to d. In contrast, with a slow-decaying eigenspectrum choosing ℓ small seems to provide relatively less improvement (these experiments are not shown). In none of our experiments does $\ell \ll d$ yield worse results compared to gradient descent when a linesearch is used, suggesting that choosing $\ell \ll d$ may be beneficial with little risk of performing worse. Further analysis must be conducted to verify this assertion. An interesting alternative to choosing a fixed ℓ is to change ℓ as the algorithm progresses in an attempt determine, locally, the appropriate dimension of the subspace used for descent. Further experimentation and analysis would be required to ascertain the benefits of such an adaptive ℓ.

4 Conclusions

We present analysis of an algorithm that generalizes Gaussian smoothing to descend in a randomly chosen subspace and have provided evidence that this generalization is appropriate for high-dimensional objective functions. We give asymptotic and non-asymptotic results of convergence under a variety of convexity assumptions. We provide tools that are useful beyond the context of this work, such as an interpretation of the Johnson–Lindenstrauss lemma that takes advantage of finite ambient dimension d. We demonstrate empirical improvements compared to the *status quo* for several practical problems, and show that the empirical performance can be good even when the assumptions required by the theory are relaxed.

The most obvious extension of this work is a generalization to the case of derivative-free optimization. With directional derivatives unavailable, finite-difference approximations of the derivatives must be employed adding a non-cancelling error at each iteration. Preliminary experiments show that this does not noticeably impede the convergence if h, the finite-difference stepsize is sufficiently small.

Thus far, analysis has only been performed for a fixed step-size, but we have shown that an adaptive step-size is required for good practical performance. Recent work in this direction [7, 13] provides promising results that may readily extend to our case. Alternatively, our analysis may be more amenable to trust region methods as in [47]

It would be interesting to adapt stochastic optimization algorithms that subsample the observations, as for example in ERM, to the stochastic subspace descent framework. Such sampling would necessitate examination into the effect that noisy function evaluations have on the convergence results. A computationally straightforward extension may allow sketching methods (see e.g. [59]) to improve our results with minimal programming overhead, but analysis must be conducted to confirm the theoretical properties of such modifications. An adaptive scheme that makes use of observed curvature information could be beneficial for determining the descent directions, an idea that has been discussed at length in the coordinate descent literature [51, 61]. Parallelizing our methods to calculate the ℓ directional derivatives at each iteration simultaneously is straightforward, but we would like to explore the feasibility of asynchronous parallelization as has been discussed in the coordinate descent case (see, e.g., [57]). Faster convergence using derivative-free quasi-Newton methods as in [5] are an obvious extension of this work. Finally, recent work on a universal "catalyst" scheme [45] also applies to our method, allowing for Nesterov-style acceleration without requiring additional knowledge of the Lipschitz constants along any particular direction.

Proofs of main results

Theorem 1

Because *f* is continuously-differentiable with a λ-Lipschitz derivative it follows that

$$f(\mathbf{x}_{k+1}) \le f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^\top(\mathbf{x}_{k+1} - \mathbf{x}_k) + \frac{\lambda}{2}\|\mathbf{x}_{k+1} - \mathbf{x}_k\|^2. \tag{18}$$

Let $f_e(\mathbf{x}) = f(\mathbf{x}) - f_*$ be the error for a particular $\mathbf{x}$. Then, (2) and (18) yield:

$$f_e(\mathbf{x}_{k+1}) - f_e(\mathbf{x}_k) \le -\alpha_\lambda \langle \nabla f(\mathbf{x}_k),\ \mathbf{P}_k\mathbf{P}_k^\top \nabla f(\mathbf{x}_k)\rangle \quad \text{with} \quad \alpha_\lambda = \alpha - d\alpha^2\lambda/(2\ell), \tag{19}$$

where we have used the fact that $\mathbf{P}_k\mathbf{P}_k^\top\mathbf{P}_k\mathbf{P}_k^\top = (d/\ell)\mathbf{P}_k\mathbf{P}_k^\top$. Any choice $0 < \alpha < 2\ell/(d\lambda)$ ensures $\alpha_\lambda > 0$. With this choice the right hand-side is non-positive and the errors are non-increasing. Since the error is bounded below by zero the sequence converges almost surely. Furthermore, since the sequence is bounded above by $f_e(\mathbf{x}_0)$, Lebesgue's dominated convergence implies convergence of the sequence in L^1. To find the actual limit, define the filtration (i.e., increasing sequence of σ-algebras) $\mathcal{F}_k = \sigma(\mathbf{P}_1, \dots, \mathbf{P}_{k-1})$, $k > 1$, and $\mathcal{F}_1 = \{\emptyset, \Omega\}$. We take conditional expectations of both sides to get

$$\mathbb{E}[f_e(\mathbf{x}_{k+1}) \mid \mathcal{F}_k] \le -\alpha_\lambda \mathbb{E}[\langle \nabla f(\mathbf{x}_k), \mathbf{P}_k\mathbf{P}_k^\top \nabla f(\mathbf{x}_k)\rangle \mid \mathcal{F}_k] + f_e(\mathbf{x}_k),$$

which leads to

$$\mathbb{E}\big(f_e(\mathbf{x}_{k+1}) \mid \mathcal{F}_k\big) \le -\alpha_\lambda \|\nabla f(\mathbf{x}_k)\|^2 + f_e(\mathbf{x}_k), \tag{20}$$

and since $\alpha_\lambda > 0$, the PL-inequality yields

$$\mathbb{E}\big(f_e(\mathbf{x}_{k+1}) \mid \mathcal{F}_k\big) \le -2\gamma\alpha_\lambda f_e(\mathbf{x}_k) + f_e(\mathbf{x}_k) = \big(1 - 2\gamma\alpha_\lambda\big) f_e(\mathbf{x}_k),$$

from which we conclude that

$$\mathbb{E} f(\mathbf{x}_{k+1}) - f_* \le \big(1 - 2\gamma\alpha_\lambda\big)^{k+1}\big(f(\mathbf{x}_0) - f_*\big).$$

Thus, since $\underset{L}{f_e}(\mathbf{x}_k) \xrightarrow{a.s.} \mathbf{x}$ for some $\mathbf{x} \in L^1$ and $f_e(\mathbf{x}_k) \xrightarrow{L^1} 0$, we have both $f(\mathbf{x}_k) \xrightarrow{a.s.} f_*$ and $f(\mathbf{x}_k) \longrightarrow f_*$.

Corollary 1(*i*)

By strong-convexity, the PL-inequality, and Theorem 1 we obtain $f(\mathbf{x}_k) \xrightarrow{a.s.} f(\mathbf{x}_*)$ and $f(\mathbf{x}_k) - f(\mathbf{x}_*) \underset{a.s.}{\geq} \frac{\gamma}{2}\|\mathbf{x}_* - \mathbf{x}_k\|$. Since the left-hand side converges a.s. to zero and $\gamma > 0$, we have $\mathbf{x}_k \longrightarrow \mathbf{x}_*$.

Corollary 1(*ii*)

Rearranging the terms in Eq. (20) we have $-\alpha_\lambda^{-1}\mathbb{E}\big(f_e(\mathbf{x}_k) - f_e(\mathbf{x}_{k+1}) \mid \mathcal{F}_k\big) \geq \|\nabla f(\mathbf{x}_k)\|^2$. Combining this with Lipschitz continuity yields $2\gamma f_e(\mathbf{x}_k) \leq \|\nabla f(\mathbf{x}_k)\|^2 \leq -\alpha_\lambda^{-1}\mathbb{E}\big(f_e(\mathbf{x}_k) - f_e(\mathbf{x}_{k+1}) \mid \mathcal{F}_k\big)$.

That is,

$$\mathbb{E}\big(f_e(\mathbf{x}_{k+1}) \mid \mathcal{F}_k\big) \leq \big(1 - 2\gamma\alpha_\lambda\big) f_e(\mathbf{x}_k). \tag{21}$$

Choosing $\alpha_\lambda = \ell/(d\lambda)$ results in $\mathbb{E}f_e(\mathbf{x}_{k+1}) \leq (1 - \ell\gamma/(d\lambda))^{k+1} f_e(\mathbf{x}_0)$ □

Theorem 2

We follow the proof of Theorem 1 until (20), then we rearrange terms to obtain,

$$\mathbb{E}\big(f(\mathbf{x}_{k+1}) \mid \mathcal{F}_k\big) \leq f(\mathbf{x}_k) - \alpha_\lambda \|\nabla f(\mathbf{x}_k)\|^2, \tag{22}$$

and then by convexity and the Cauch-Schwarz inequality, $\|\nabla f(\mathbf{x}_k)\| \geq f_e(\mathbf{x}_k)/R$. Plugging this into Eq. (22) and letting $\alpha = \ell/(d\lambda)$ results in

$$\mathbb{E}[f_e(\mathbf{x}_{k+1}) \mid \mathcal{F}_k] - f_e(\mathbf{x}_k) \leq -\alpha f_e(\mathbf{x}_k)^2/2R^2, \tag{23}$$

and one more expectation yields

$$\begin{aligned}\mathbb{E}[f_e(\mathbf{x}_{k+1}) - f_e(\mathbf{x}_k)] &\leq -\alpha \mathbb{E}f_e(\mathbf{x}_k)^2/2R^2 \leq -\alpha\big(\mathbb{E}f_e(\mathbf{x}_k)\big)^2/2R^2 \\ &\leq -\alpha \mathbb{E}f_e(\mathbf{x}_k) \cdot \mathbb{E}f_e(\mathbf{x}_{k+1})/(2R^2)\end{aligned}$$

since $\alpha \geq 0$ and $\mathbb{E}f_e(\mathbf{x}_{k+1}) \leq \mathbb{E}f_e(\mathbf{x}_k)$. Dividing by $\mathbb{E}f_e(\mathbf{x}_k) \cdot \mathbb{E}f_e(\mathbf{x}_{k+1})$ gives

$$\frac{1}{\mathbb{E}f_e(\mathbf{x}_{k+1})} \geq \frac{1}{\mathbb{E}f_e(\mathbf{x}_k)} + \frac{\alpha}{2R^2}. \tag{24}$$

Applying (24) recursively, and replacing α with $\ell/(d\lambda)$ we obtain $\mathbb{E}f_e(\mathbf{x}_{k+1}) \leq 2d\lambda R^2/k\ell$.

Theorem 3

Beginning from (20) we set $\alpha_\lambda = \ell/(d\lambda)$ and rearrange terms to get

$$\ell/(2d\lambda)\left\|\nabla f(\mathbf{x}_k)\right\|^2 \leq f(\mathbf{x}_k) - \mathbb{E}(f(\mathbf{x}_{k+1}) \mid \mathcal{F}_k),$$

which leads to

$$\ell/(2d\lambda)\sum_{i=0}^{k}\mathbb{E}\left\|\nabla f(\mathbf{x}_k)\right\|^2 \leq \sum_{i=0}^{k}\mathbb{E}(f(\mathbf{x}_i) - f(\mathbf{x}_{i+1})) = f(\mathbf{x}_0) - \mathbb{E}f(\mathbf{x}_{k+1}) \leq f(\mathbf{x}_0) - f_*.$$

Recognizing that a sum of $k+1$ values is bounded below by $k+1$ replicates of its minimum yields

$$(k+1)\min_{i\in\{0,\ldots,k\}}\mathbb{E}\left\|\nabla f(\mathbf{x}_k)\right\|^2 \leq \frac{2d\lambda(f(\mathbf{x}_0) - f_*)}{\ell}.$$

Divide both sides by $k+1$ to get the result. Now, define some tolerance ϵ such that

$$\frac{2d\lambda(f(\mathbf{x}_0) - f_*)}{(k+1)\ell} \leq \epsilon.$$

Then,

$$k \geq \frac{2d\lambda(f(\mathbf{x}_0) - f_*)}{\epsilon\ell} - 1.$$

That is, $k = \mathcal{O}(d/\ell\epsilon)$ iterations are sufficient to achieve $\mathbb{E}\left\|\nabla f(\mathbf{x}_k)\right\| \leq \epsilon$.

Lemma 1

Let $\mathbf{H} \in \mathbb{R}^{d\times d}$ be a Haar-distributed random matrix, $\mathbf{v} \in \mathbb{R}^d$ an arbitrary fixed vector, and $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$. Then $\mathbf{H}^\top\mathbf{v}/\|\mathbf{v}\|$ and $\mathbf{u}/\|\mathbf{u}\|$ are both distributed uniformly on the d-dimensional sphere. Let $\mathbf{I}_{\ell\times d} \in \mathbb{R}^{\ell\times d}$ represent a mapping onto the first ℓ coordinates. Then,

$$\left\|\mathbf{I}_{\ell\times d}\mathbf{u}\right\|^2 = \left(u_1^2 + \ldots + u_\ell^2\right) \sim \chi^2(\ell),$$

and

$$\|\mathbf{u}\|^2 = u_1^2 + \ldots + u_\ell^2 + u_{\ell+1}^2 + \ldots + u_d^2 \sim \chi^2(d).$$

For independent random variables $X \sim \chi^2(\alpha)$ and $Y \sim \chi^2(\beta)$, $Z = X/(X+Y) \sim \mathcal{B}\mathrm{eta}(\alpha/2, \beta/2)$. Thus,

$$\frac{\|\mathbf{I}_{\ell\times d}\mathbf{H}^\top\mathbf{v}\|^2}{\|\mathbf{v}\|^2} = \left\|\mathbf{I}_{\ell\times d}\mathbf{H}^\top\frac{\mathbf{v}}{\|\mathbf{v}\|}\right\|^2 \overset{d}{=} \left\|\mathbf{I}_{\ell\times d}\frac{\mathbf{u}}{\|\mathbf{u}\|}\right\|^2 = \frac{\|\mathbf{I}_{\ell\times d}\mathbf{u}\|^2}{\|\mathbf{u}\|^2} \sim \mathcal{B}\mathrm{eta}(\ell/2, (d-\ell)/2).$$

By construction, $\mathbf{P}_k \overset{d}{=} \sqrt{d/\ell}\,\mathbf{I}_{\ell\times d}\mathbf{H}$, so

$$\mathbb{P}\left(\left\|\mathbf{P}_k^\top \mathbf{v}\right\|^2 \le (1-\epsilon)\|\mathbf{v}\|^2\right) = \mathbb{P}\left(\left\|\mathbf{I}_{\ell\times d}\mathbf{H}^\top \frac{\mathbf{v}}{\|\mathbf{v}\|}\right\|^2 \le \frac{\ell}{d}(1-\epsilon)\right).$$

The Beta CDF is calculated by evaluating the regularized incomplete Beta function. That is, if $X \sim \mathcal{B}eta(\alpha,\beta)$ then $F_X(p) = I_p(\alpha,\beta)$. Thus, the probability

$$\mathbb{P}\left(\left\|\mathbf{I}_{\ell\times d}\mathbf{H}^\top \frac{\mathbf{v}}{\|\mathbf{v}\|}\right\|^2 \ge \frac{\ell}{d}(1-\epsilon)\right) = 1 - I_{(1-\epsilon)\ell/d}(\ell/2, (d-\ell)/2)$$

provides a probability of a successful embedding.

Remark 3

We show that for any $k_1 < \cdots < k_m$ the sets $A_{k_1}, \ldots, A_{k_m}$ are mutually independent. Let $\mathbb{1}_A$ denote the indicator function of a set A. Define the filtration (i.e., increasing sequence of σ-algebras) $\mathcal{F}_k = \sigma(\mathbf{P}_1, \ldots, \mathbf{P}_{k-1})$, $k > 1$, and $\mathcal{F}_1 = \{\emptyset, \Omega\}$ and note that A_k is $\mathcal{F}_{k-1}$–measurable. Then by the chain rule of probability and the fact that the $(\mathbf{P}_k)$ are iid,

$$\begin{aligned}
\mathbb{P}(A_{k_1} \cap \cdots \cap A_{k_m}) &= \mathbb{E}[\mathbb{1}_{A_{k_1}} \cdots \mathbb{1}_{A_{k_{m-1}}}]\mathbb{P}(A_{k_m} \mid \mathcal{F}_{k_{m-1}}) \\
&= \mathbb{E}[\mathbb{1}_{A_{k_1}} \cdots \mathbb{1}_{A_{k_{m-2}}}]\mathbb{P}(A_{k_{m-1}} \mid \mathcal{F}_{k_{m-2}})\mathbb{P}(A_{k_m} \mid \mathcal{F}_{k_{m-1}}) \\
&\vdots \\
&= \mathbb{P}(A_{k_1} \mid \mathcal{F}_{k_0}) \cdots \mathbb{P}(A_{k_m} \mid \mathcal{F}_{k_{m-1}}) \\
&= \mathbb{P}(A_{k_1}) \cdots \mathbb{P}(A_{k_m}).
\end{aligned}$$

Theorem 4

Beginning from (19) we choose an ℓ determined by Lemma 1 such that with probability δ,

$$f_e(\mathbf{x}_k) \le f_e(\mathbf{x}_{k-1}) - (1-\epsilon)\alpha_\lambda \left\|\nabla f(\mathbf{x}_{k-1})\right\|^2. \tag{25}$$

By (A3') the function is γ-strongly-convex, so,

$$f_e(\mathbf{x}_k) \le \left(1 - (1-\epsilon)\frac{\ell\gamma}{d\lambda}\right) f_e(\mathbf{x}_{k-1}) \quad \text{with probability } \delta. \tag{26}$$

Define the Bernoulli random variable $W_k \sim Bern(\delta)$ such that $W_k = 1$, occurring with probability δ, constitutes a successful embedding on the k^{th} iteration. We can re-write (26) as

$$f_e(\mathbf{x}_k) \le \big(1 - W_k(1-\omega)\big) f_e(\mathbf{x}_{k-1}), \tag{27}$$

where $\omega = 1 - (1-\epsilon)\ell\gamma/(d\lambda)$. If the embedding is a failure, we use the trivial bound $\left\|\mathbf{P}_k^\top \nabla f(\mathbf{x}_k)\right\|^2 \geq 0$. Consider a random variable $U_k = 1 - W_k(1-\omega)$, then (27) is

$$f_e(\mathbf{x}_k) \leq (U_1 \cdots U_k) f_e(\mathbf{x}_0).$$

Note that $\log(U_k) = Y_k \log \omega$ for $Y_k \sim \text{Bernoulli}(\delta)$. Let $B \sim \text{Bin}(k, \delta)$, then, for $t' \in (0,\, k\delta]$

$$\mathbb{P}(U_1 \cdots U_k \geq \omega^{k\delta - t'}) = \mathbb{P}(B \log \omega \geq (k\delta - t') \log \omega) = \mathbb{P}(B \leq k\delta - t'). \quad (28)$$

Thus, for $t' \in (0, k\delta]$ we obtain a probabilistic lower bound on the improvement using Remark 2,

$$\mathbb{P}(U_1 \cdots U_k \geq \omega^{k\delta - t'}) \leq \exp(-t'^2/2\sigma_k^2),$$

where $\sigma_k^2 = k(1-2\delta)/(2\log((1-\delta)/\delta))$. Now,

$$\begin{aligned}\mathbb{P}\left(f_e(\mathbf{x}_k) \geq \omega^{k\delta - t'}\right) &\leq \mathbb{P}\left((U_1 \cdots U_k) f_e(\mathbf{x}_0) \geq \omega^{k\delta - t'}\right) \\ &= \mathbb{P}((U_1 \cdots U_k) \geq \omega^{k\delta - t'}/f_e(\mathbf{x}_0))\end{aligned}$$

which implies that for $t' \in (0,\, k\delta]$,

$$\mathbb{P}\left(f_e(\mathbf{x}_k) \geq \left(1 - (1-\epsilon)\frac{\ell\gamma}{d\lambda}\right)^{k\delta - t'} f_e(\mathbf{x}_0)\right) \leq \exp(-t'^2/2\sigma_k^2).$$

Define $t = (t'/k) \in (0, \delta]$ and the result follows.

Acknowledgements We thank Lior Horesh for his suggestions regarding derivative-free optimization, Gregory Fasshauer for fruitful discussions on sparse Gaussian processes, and Neil Longfellow and Osman Malik for their insights into connections with the Johnson–Lindenstrauss lemma. We also thank Subhayan De for providing the FEniCS code used in Sect. 3.3. AD acknowledges funding by the US Department of Energy's Office of Science Advanced Scientific Computing Research, Award DE-SC0006402 and National Science Foundation Grant CMMI-145460. LT acknowledges funding by National Science Foundation Grant DMS-1723005. SB acknowledges funding by National Science Foundation Grant DMS-1819251.

References

1. Abacioglu, Y., Oliver, D., Reynolds, A.: Efficient reservoir history matching using subspace vectors. Comput. Geosci. **5**, 151–172 (2001)
2. Achlioptas, D.: Database-friendly random projections: Johnson–Lindenstrauss with binary coins. J. Comput. Syst. Sci. **66**, 671–687 (2003)
3. Allen-Zhu, Z., Qu, Z., Richtárik, P., Yuan, Y.: Even faster accelerated coordinate descent using non-uniform sampling. In: ICML, pp. 1110–1119 (2016)
4. Arbel, J., Marchal, O., Nguyen, H.D.: On strict sub-Gaussianity, optimal proxy variance and symmetry for bounded random variables. ESAIM Probab. Stat. **24**, 39–55 (2020)
5. Berahas, A.S., Byrd, R.H., Nocedal, J.: Derivative-free optimization of noisy functions via quasi-Newton methods. SIAM J. Optim. **29**, 965–993 (2019)

6. Berahas, A.S., Cao, L., Choromanski, K., Scheinberg, K.: A theoretical and empirical comparison of gradient approximations in derivative-free optimization. arXiv preprint arXiv:1905.01332 (2019)
7. Berahas, A.S., Cao, L., Scheinberg, K.: Global convergence rate analysis of a generic line search algorithm with noise (2019)
8. Bertsimas, D., Vempala, S.: Solving convex programs by random walks. J. ACM **51**, 540–556 (2004)
9. Bjarkason, E.K., Maclaren, O.J., O'Sullivan, J.P., O'Sullivan, M.J.: Randomized truncated SVD Levenberg–Marquardt approach to geothermal natural state and history matching. Water Resour. Res. **54**, 2376–2404 (2018)
10. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. SIAM Rev. **60**, 223–311 (2018)
11. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
12. Bui-Thanh, T., Ghattas, O., Martin, J., Stadler, G.: A computational framework for infinite-dimensional Bayesian inverse problems part I: the linearized case, with application to global seismic inversion. SIAM J. Sci. Comput. **35**, A2494–A2523 (2013)
13. Cartis, C., Scheinberg, K.: Global convergence rate analysis of unconstrained optimization methods based on probabilistic models. Math. Program. **169**, 337–375 (2018)
14. Choromanski, K., Rowland, M., Sindhwani, V., Turner, R.E., Weller, A.: Structured evolution with compact architectures for scalable policy optimization. In: ICML (2018)
15. Conn, A.R., Scheinberg, K., Vicente, L.N.: Introduction to Derivative-Free Optimization, vol. 8. SIAM, Philadelphia (2009)
16. Cui, T., Martin, J., Marzouk, Y.M., Solonen, A., Spantini, A.: Likelihood-informed dimension reduction for nonlinear inverse problems. Inverse Prob. **30**, 114015 (2014)
17. Dabbene, F., Shcherbakov, P.S., Polyak, B.T.: A randomized cutting plane method with probabilistic geometric convergence. SIAM J. Optim. **20**, 3185–3207 (2010)
18. De, S., Maute, K., Doostan, A.: Bi-fidelity stochastic gradient descent for structural optimization under uncertainty. arXiv preprint arXiv:1911.10420 (2019)
19. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)
20. Drori, Y., Teboulle, M.: Performance of first-order methods for smooth convex minimization: a novel approach. Math. Program. **145**, 451–482 (2014)
21. Duchi, J.C., Jordan, M.I., Wainwright, M.J., Wibisono, A.: Optimal rates for zero-order convex optimization: the power of two function evaluations. IEEE Trans. Inf. Theory **61**, 2788–2806 (2015)
22. Dvurechensky, P., Gasnikov, A., Gorbunov, E.: An accelerated directional derivative method for smooth stochastic convex optimization. arXiv preprint arXiv:1804.02394 (2018)
23. Dvurechensky, P., Gasnikov, A., Tiurin, A.: Randomized similar triangles method: a unifying framework for accelerated randomized optimization methods (coordinate descent, directional search, derivative-free method). arXiv preprint arXiv:1707.08486 (2017)
24. Ermoliev, Y., Wets, R.-B.: Numerical Techniques for Stochastic Optimization. Springer, Berlin (1988)
25. Flath, H., Wilcox, L., Akçelik, V., Hill, J., Van Bloemen Waanders, B., Ghattas, O.: Fast algorithms for Bayesian uncertainty quantification in large-scale linear inverse problems based on low-rank partial hessian approximations. SIAM J. Sci. Comput. **33**, 407–432 (2011)
26. Frankl, P., Maehara, H.: Some geometric applications of the beta distribution. Ann. Inst. Stat. Math. **42**, 463–474 (1990)
27. Gaviano, M.: Some general results on convergence of random search algorithms in minimization problems. Towards Glob. Optim. 149–157 (1975)
28. Ge, R., Huang, F., Jin, C., Yuan, Y.: Escaping from saddle points: online stochastic gradient for tensor decomposition. In: Conference on Learning Theory, pp. 797–842 (2015)
29. Ghadimi, S., Lan, G.: Stochastic first- and zeroth-order methods for nonconvex stochastic programming. SIAM J. Optim. **23**(1), 2341–2368 (2013)
30. Gower, R.M., Richtárik, P.: Stochastic dual ascent for solving linear systems. arXiv preprint arXiv:1512.06890 (2015)
31. Griewank, A., Walther, A.: Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, vol. 105, 2nd edn. SIAM, Philadelphia (2008)
32. Gunzburger, M.D.: Perspectives in Flow Control and Optimization, vol. 5. SIAM, Philadelphia (2003)

33. Haber, E., Chung, M., Herrmann, F.: An effective method for parameter estimation with PDE constraints with multiple right-hand sides. SIAM J. Optim. **22**, 739–757 (2012)
34. Haber, E., Magnant, Z., Lucero, C., Tenorio, L.: Numerical methods for A-optimal designs with a sparsity constraint for ill-posed inverse problems. Comput. Optim. Appl. **52**, 293–314 (2012)
35. Hager, W.W., Zhang, H.: Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent. ACM Trans. Math. Softw. **32**, 113–137 (2006)
36. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evol. Comput. **9**, 159–195 (2001)
37. Hanzely, F., Richtárik, P.: Accelerated coordinate descent with arbitrary sampling and best rates for minibatches. arXiv preprint arXiv:1809.09354 (2018)
38. Horesh, L., Haber, E., Tenorio, L.: Optimal experimental design for the large-scale nonlinear ill-posed problem of impedance imaging. In: Biegler, L., et al. (eds.) Large-Scale Inverse Problems and Quantification of Uncertainty. Wiley Series in Computational Statistics, pp. 273–290. Wiley, Chichester (2010)
39. Hua, X., Yamashita, N.: Iteration complexity of a block coordinate gradient descent method for convex optimization. SIAM J. Optim. **25**, 1298–1313 (2015)
40. Isaac, T., Petra, N., Stadler, G., Ghattas, O.: Scalable and efficient algorithms for the propagation of uncertainty from data through inference to prediction for large-scale problems, with application to flow of the Antarctic ice sheet. J. Comput. Phys. **296**, 348–368 (2015)
41. Johnson, R., Zhang, T.: Accelerating stochastic gradient descent using predictive variance reduction. NIPS **26**, 315–323 (2013)
42. Kimeldorf, G.S., Wahba, G.: A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. Ann. Math. Stat. **41**, 495–502 (1970)
43. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**, 671–680 (1983)
44. Leventhal, D., Lewis, A.: Randomized Hessian estimation and directional search. Optimization **60**, 329–345 (2011)
45. Lin, H., Mairal, J., Harchaoui, Z.: A universal catalyst for first-order optimization. NIPS **28**, 3384–3392 (2015)
46. Logg, A., Mardal, K.-A., Wells, G.: Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book, vol. 84. Springer, Berlin (2012)
47. Maggiar, A., Wächter, A., Dolinskaya, I.S., Staum, J.: A derivative-free trust-region algorithm for the optimization of functions smoothed via Gaussian convolution using adaptive multiple importance sampling. SIAM J. Optim. **28**, 1478–1507 (2018)
48. Marchal, O., Arbel, J., et al.: On the sub-Gaussianity of the beta and Dirichlet distributions. Electron. Commun. Probab. **22** (2017). https://doi.org/10.1214/17-ECP92
49. Mezzadri, F.: How to generate random matrices from the classical compact groups. In: Notices of the American Mathematical Society, vol. 54 (2006)
50. Nesterov, Y.: A method of solving a convex programming problem with convergence rate $O(1/k^2)$. Sov. Math. Dokl. **27**, 372–376 (1983)
51. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM J. Optim. **22**, 341–362 (2012)
52. Nesterov, Y.: Introductory Lectures on Convex Optimization: A Basic Course, vol. 87. Springer, Berlin (2013)
53. Nesterov, Y., Spokoiny, V.: Random gradient-free minimization of convex functions. Found. Comput. Math. **17**, 527–566 (2017). First appeared as CORE discussion paper 2011
54. Nielsen, E.J., Diskin, B.: Discrete adjoint-based design for unsteady turbulent flows on dynamic overset unstructured grids. AIAA J. **51**, 1355–1373 (2013)
55. Nocedal, J., Wright, S.: Numerical Optimization, 2nd edn. Springer, Berlin (1999)
56. Paquette, C., Scheinberg, K.: A stochastic line search method with expected complexity analysis. SIAM J. Optim. **30**, 349–376 (2020)
57. Peng, Z., Xu, Y., Yan, M., Yin, W.: Arock: an algorithmic framework for asynchronous parallel coordinate updates. SIAM J. Sci. Comput. **38**, A2851–A2879 (2016)
58. Petra, N., Martin, J., Stadler, G., Ghattas, O.: A computational framework for infinite-dimensional Bayesian inverse problems, part ii: Stochastic Newton MCMC with application to ice sheet flow inverse problems. SIAM J. Sci. Comput. **36**, A1525–A1555 (2014)
59. Pilanci, M., Wainwright, M.J.: Randomized sketches of convex programs with sharp guarantees. IEEE Trans. Inf. Theory **61**, 5096–5115 (2015)

60. Powell, M.J.: On search directions for minimization algorithms. Math. Program. **4**, 193–201 (1973)
61. Richtárik, P., Takác, M.: Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. Math. Program. **144**, 1–38 (2014)
62. Shalev-Shwartz, S., Zhang, T.: Stochastic dual coordinate ascent methods for regularized loss minimization. JMLR **14**, 567–599 (2013)
63. Snelson, E., Ghahramani, Z.: Sparse Gaussian processes using pseudo-inputs. NIPS **18**, 1257–1264 (2006)
64. Solis, F., Wets, R.J.-B.: Minimization by random search techniques. Math. Oper. Res. **6**, 19–30 (1981)
65. Stich, S.U., Muller, C., Gartner, B.: Optimization of convex functions with random pursuit. SIAM J. Optim. **23**, 1284–1309 (2013)
66. Titsias, M.: Variational learning of inducing variables in sparse Gaussian processes. In: AISTATS, pp. 567–574 (2009)
67. Wang, Q., Moin, P., Iaccarino, G.: Minimal repetition dynamic checkpointing algorithm for unsteady adjoint calculation. SIAM J. Sci. Comput. **31**, 2549–2567 (2009)
68. Warga, J.: Minimizing certain convex functions. J. Soc. Ind. Appl. Math. **11**, 588–593 (1963)
69. Williams, C.K., Rasmussen, C.E.: Gaussian Processes for Machine Learning, vol. 2. MIT Press, Cambridge (2006)
70. Williams, C.K., Seeger, M.: Using the Nyström method to speed up kernel machines. NIPS **14**, 682–688 (2001)
71. Wright, S.J.: Coordinate descent algorithms. Math. Program. **151**, 3–34 (2015)